Efficient Online and Batch Learning using Forward Backward Splitting

J. Duchi, Y. Singer

Journal of Machine Learning Research, 2009

中田研究室

12M42340　張　志鋒

What is Machine Learning

Regression 回帰問題

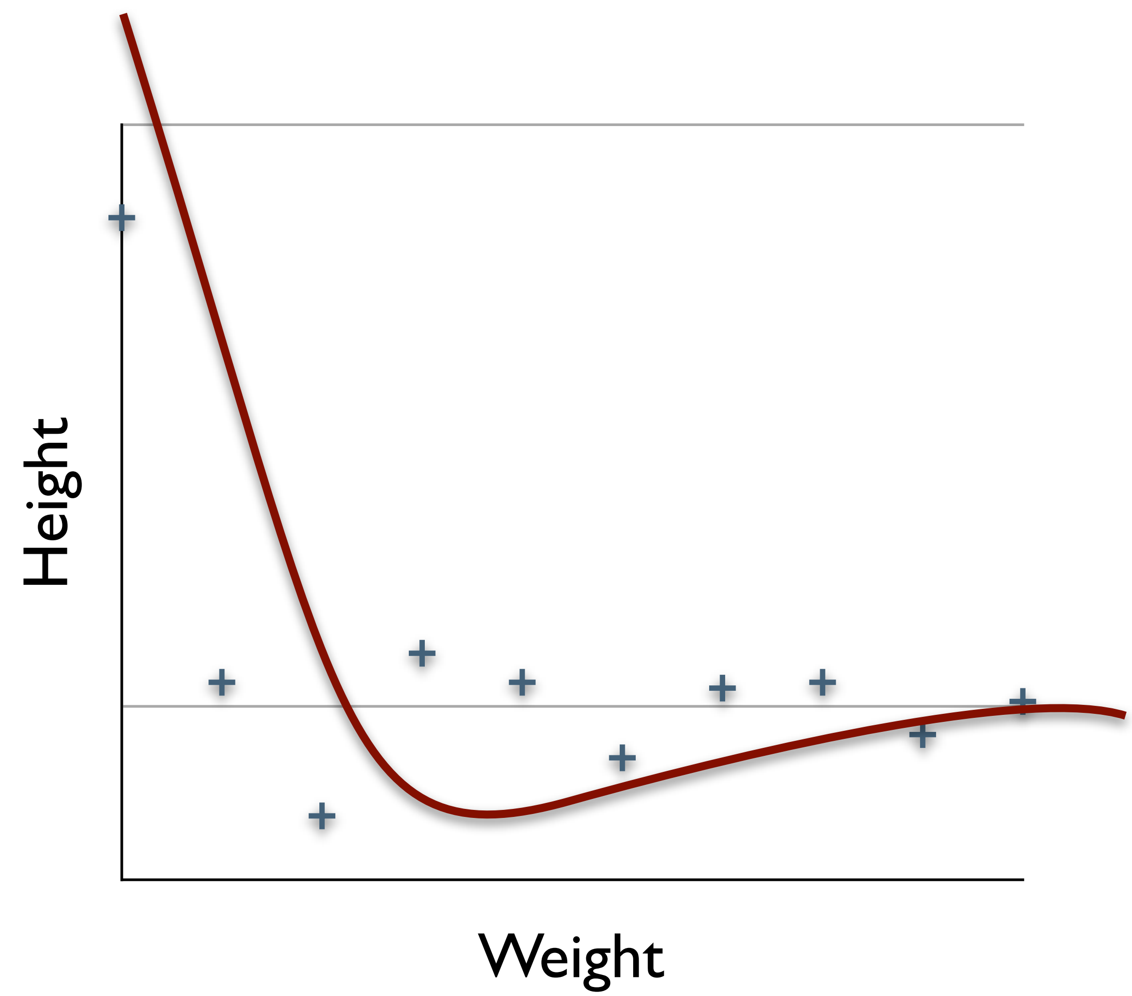
Classification 分類問題

What is Machine Learning

$X \in \mathbb{R}^n \times 1$ $\qquad$ $Y \in \mathbb{R}^n \times 1$

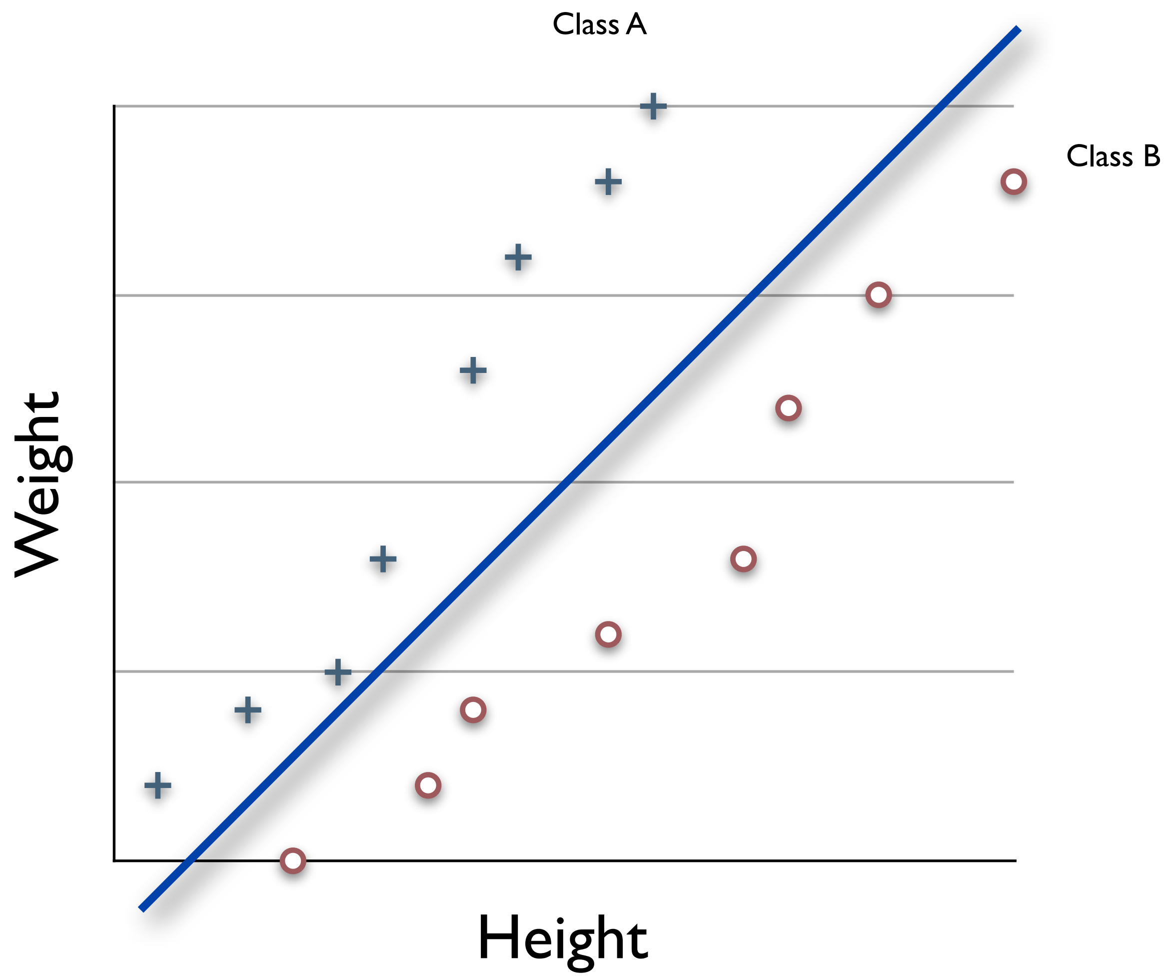
Weight	Height
1	5
4	0
2	5
7	2
8	4
...	...
4	?

Prediction



Regression

What is Machine Learning



Classification

$X \in \mathbb{R}^n \times 2$

$Y \in \{-1, 1\}^n$

Weight	Height	Class
1	5	A
4	0	B
2	5	A
7	2	B
8	4	B
...	...	...
4	6	?

Prediction

Formulation of Machine Learning

Classification

Regression

Loss function

$$\Rightarrow \quad \min \ L(w) + R(w)$$

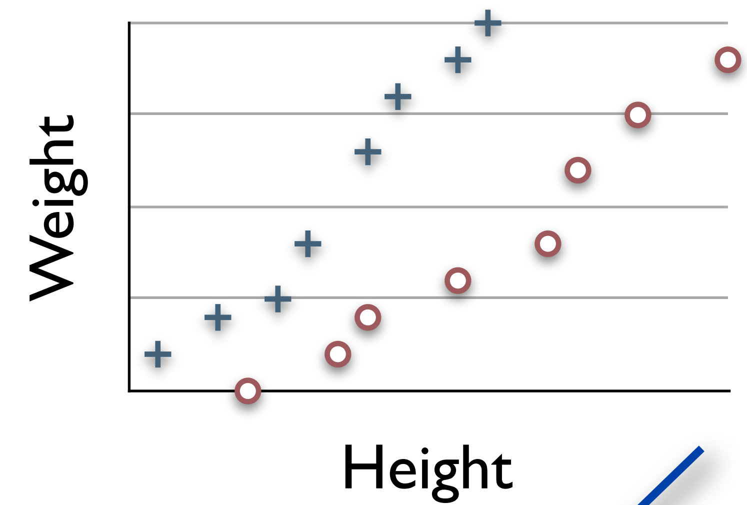
Regularizer

The Goal is to minimize this loss function with regularizer, yielding a regularized sparse solution

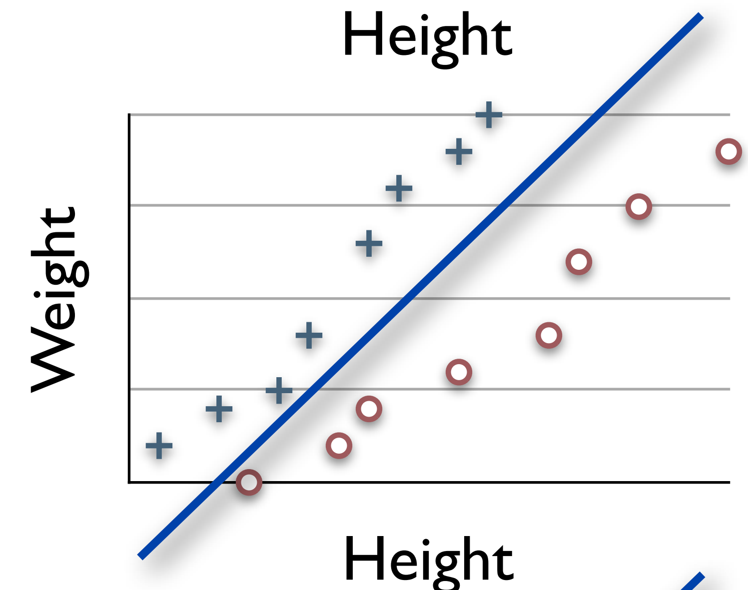
Learning

Weight	Height	Class
1	5	A
4	0	B
2	5	A
7	2	B
8	4	B

...

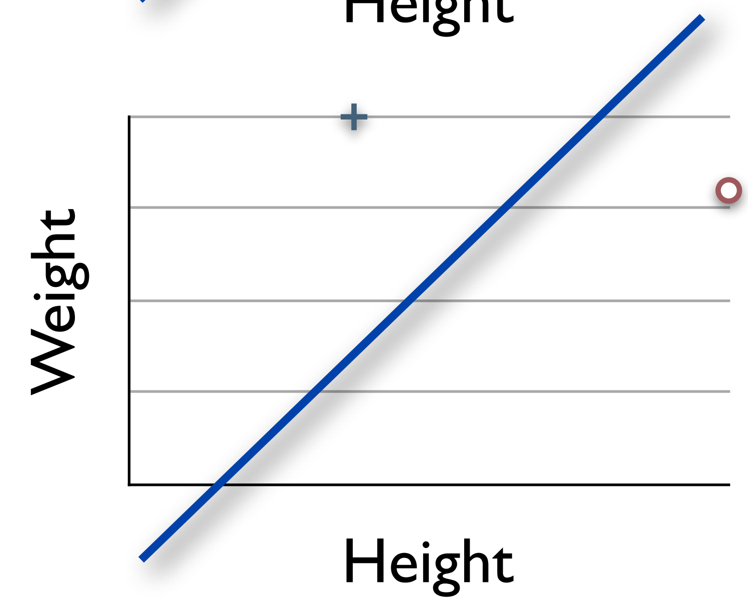


$$\min L(w) + R(w) \ \ \text{find out}\ w^*$$



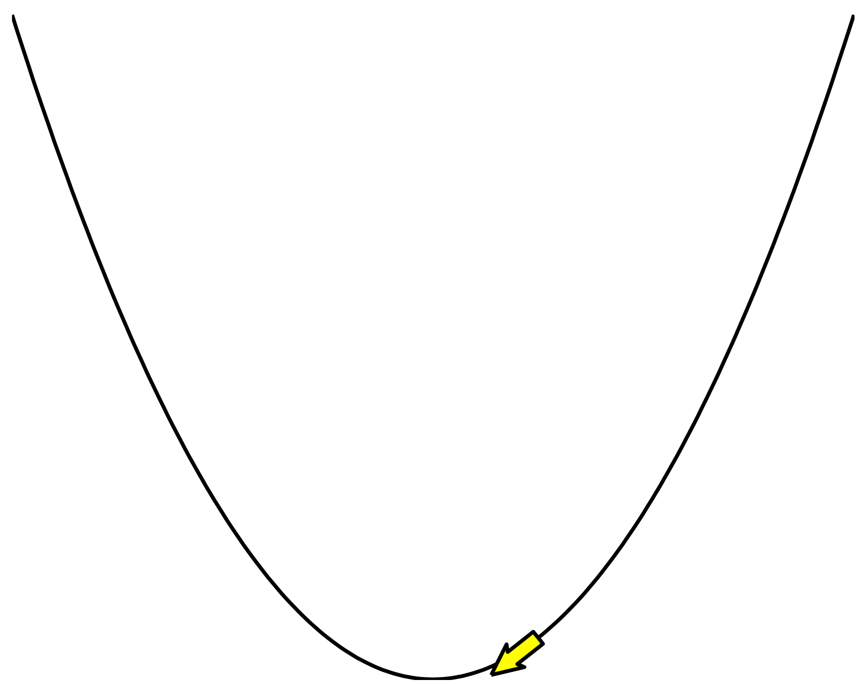
5	3	A ? B
8	4	A ? B

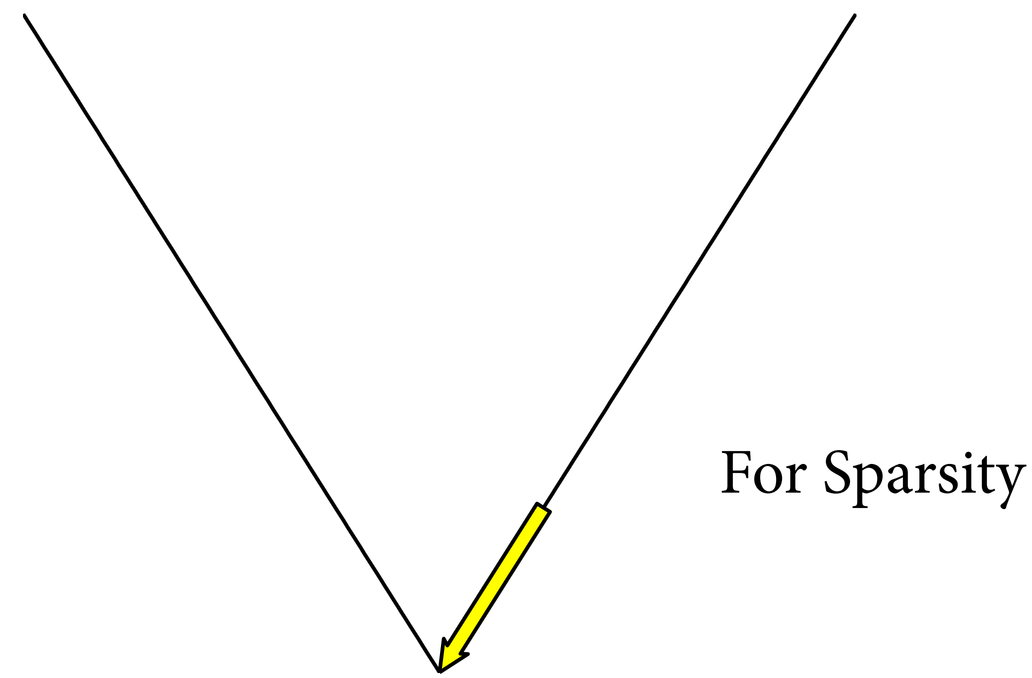
...



What is the problem?

$L(w)$ $\min \; L(w) + R(w)$ $R(w)$





Convex &
Differentiable

凸かつ微分可能

Convex but
Non-differentiable

凸かつ微分不可能

What is the problem?

A **sparse matrix** is a matrix populated primarily with **zeros**

$$w^* = \begin{pmatrix} 1 \\ 0.2 \\ 9 \\ 0 \\ 0.7 \\ 1 \end{pmatrix} \qquad \bar{w} = \begin{pmatrix} 2 \\ 0 \\ 10 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Accurate
正確さ

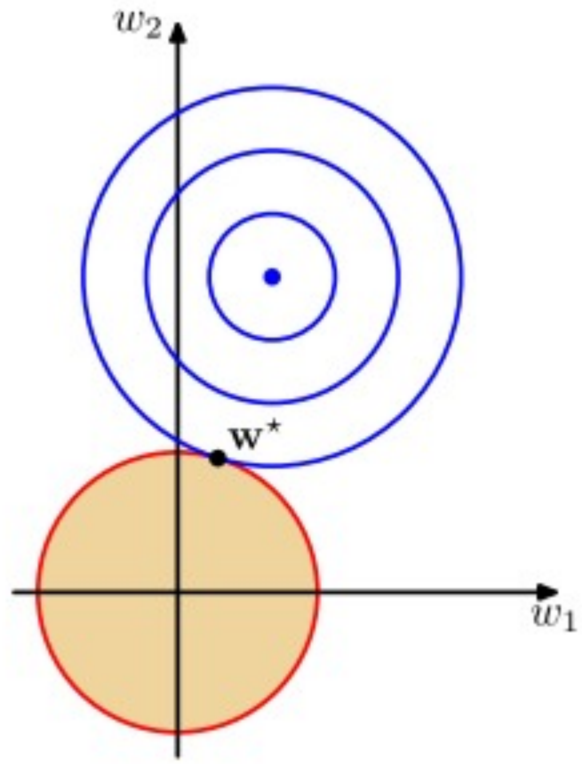
Trade-off

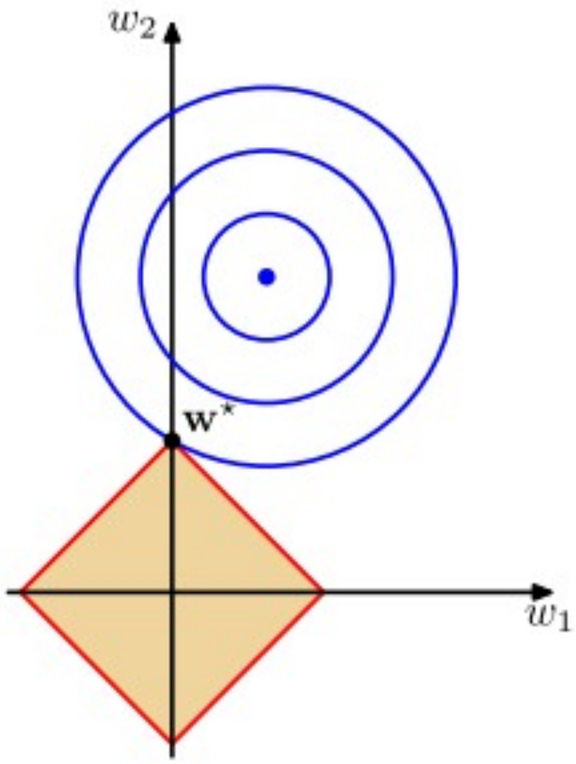
Sparse
疎性

In machine learning, we prefer sparse solutions

What is the problem?

$$w = \begin{pmatrix} 0.110 \\ 0.994 \end{pmatrix}$$





$$w = \begin{pmatrix} 0.000 \\ 1.000 \end{pmatrix}$$

$$\ell_1\text{-norm: } |w|_1 = |w^{(1)}| + |w^{(2)}| + \cdots + |w^{(3)}|$$

ℓ_1-norm lead to sparse solutions

Approaches of Machine Learning

Batch learning

Online learning

Batch Learning

Weight	Height	Class
1	5	A
4	0	B
2	5	A
7	2	B
8	4	B

…

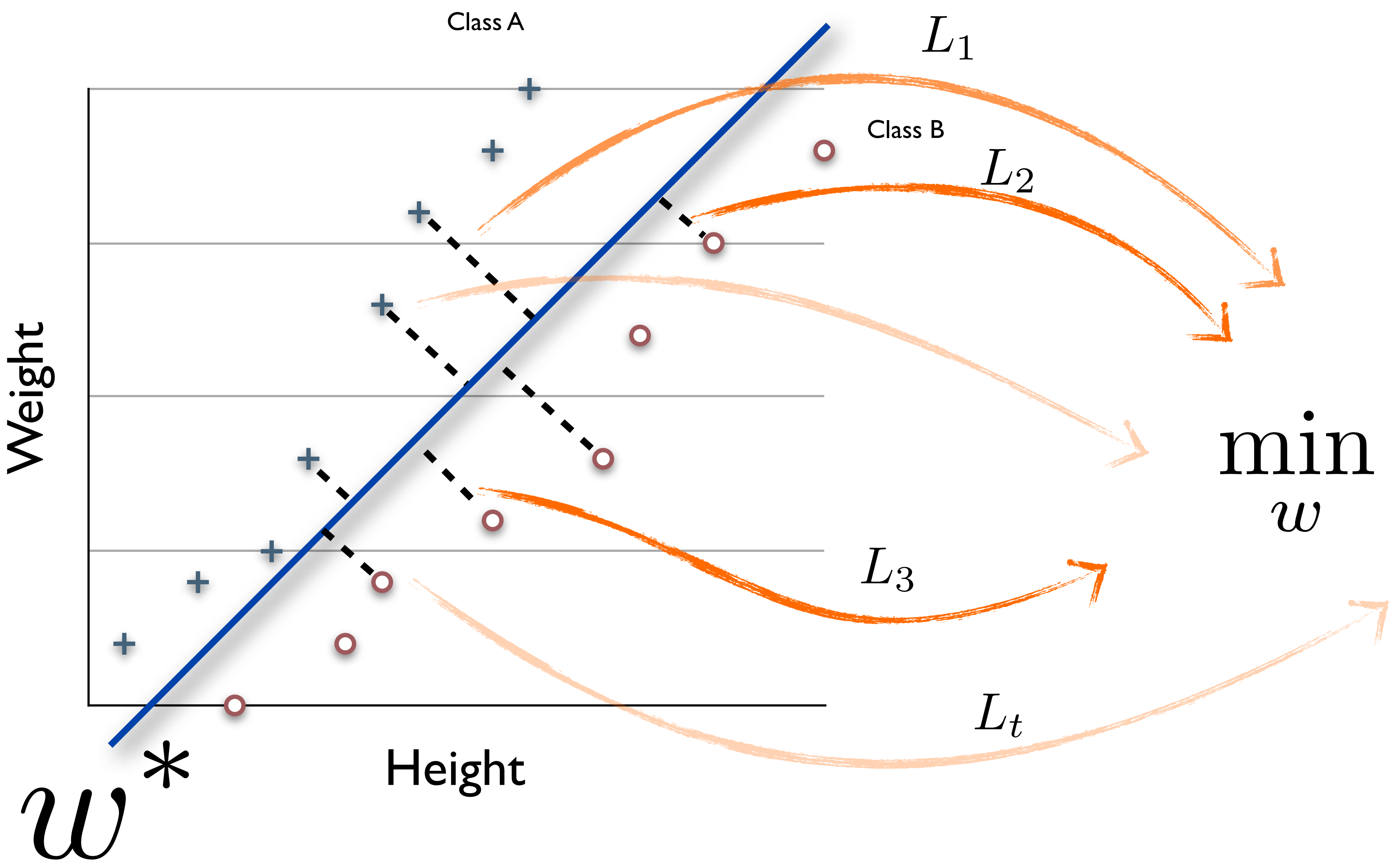
5	3	A ? B
8	4	A ? B

$$\min \; L(w) + R(w)$$

Find out w^*

Use w^* to predict new data

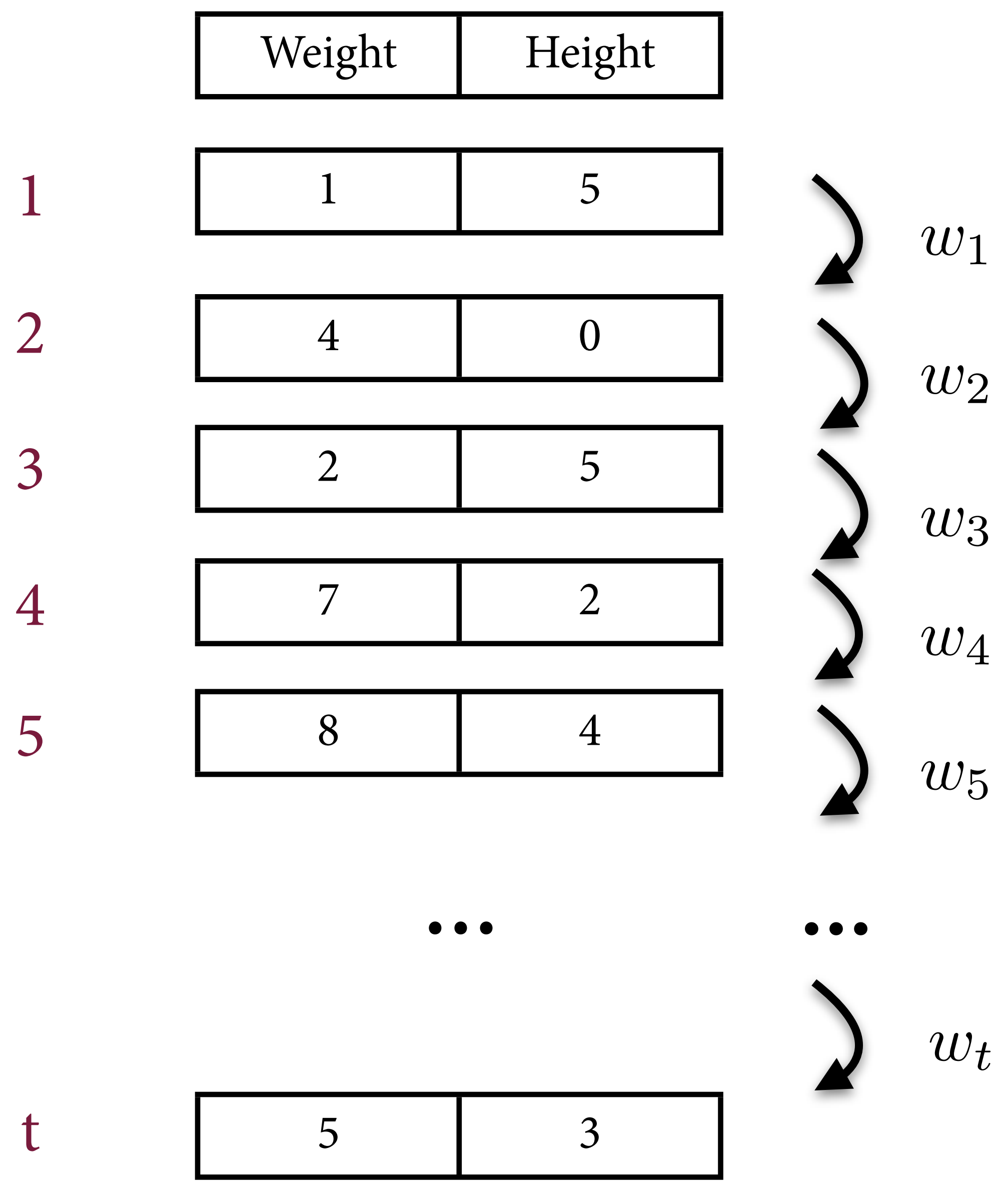
What is Batch Learning



Find out w^*

$$\min_w \sum_t L_t(w)$$

What is Online Learning

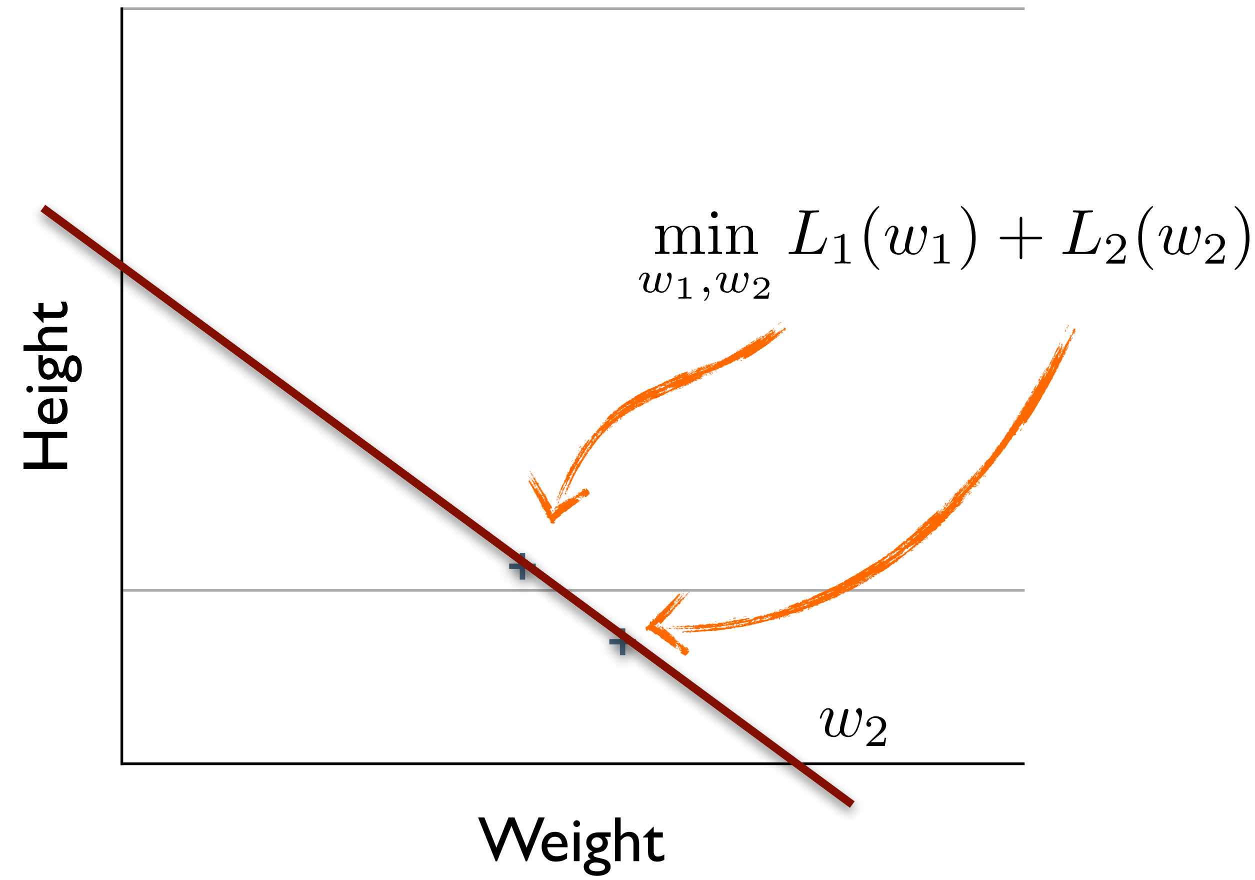


Calculate w_t,
use w_t to predict new data,
then update w_{t+1}

What is Online Learning

w_1

Add Data 2 $\longrightarrow$ w_2



Regression

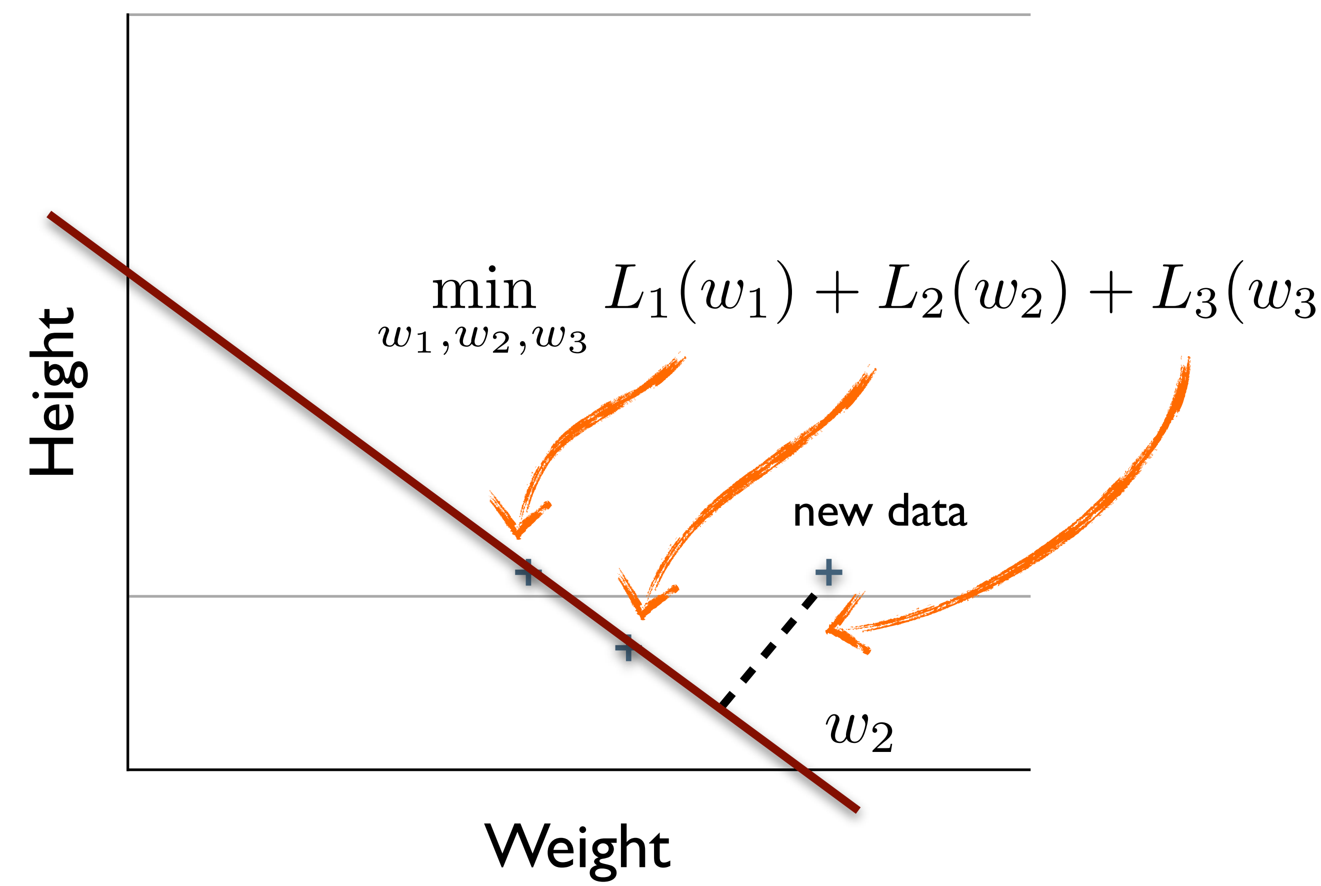
What is Online Learning

w_1

Add Data 2 $\longrightarrow$ w_2

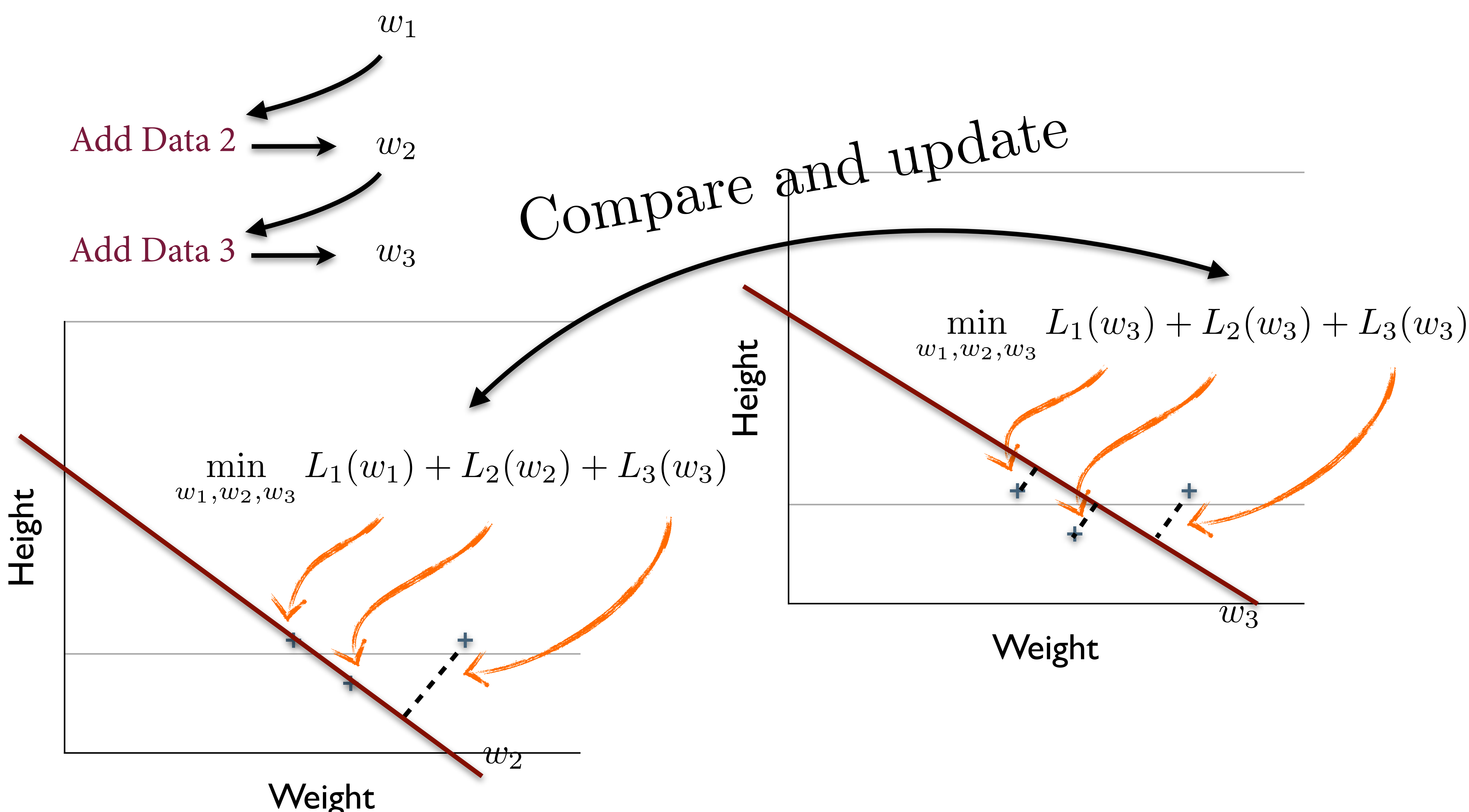
Add Data 3 $\longrightarrow$ w_3

Try to predict w_3



Regression

What is Online Learning



What is Online Learning

w_1

Add Data 2 $\longrightarrow$ w_2

Add Data 3 $\longrightarrow$ w_3

Add Data 4 $\longrightarrow$ w_4

Add Data 5 $\longrightarrow$ w_5

...

w_{t-1}

Add Data t $\longrightarrow$ w_t

Online Learning

$$\min_{w_1, w_2, \ldots, w_{t-1}} \sum_t L_t(w_t)$$

Batch Learning

$$\min_w \sum_t L_t(w)$$

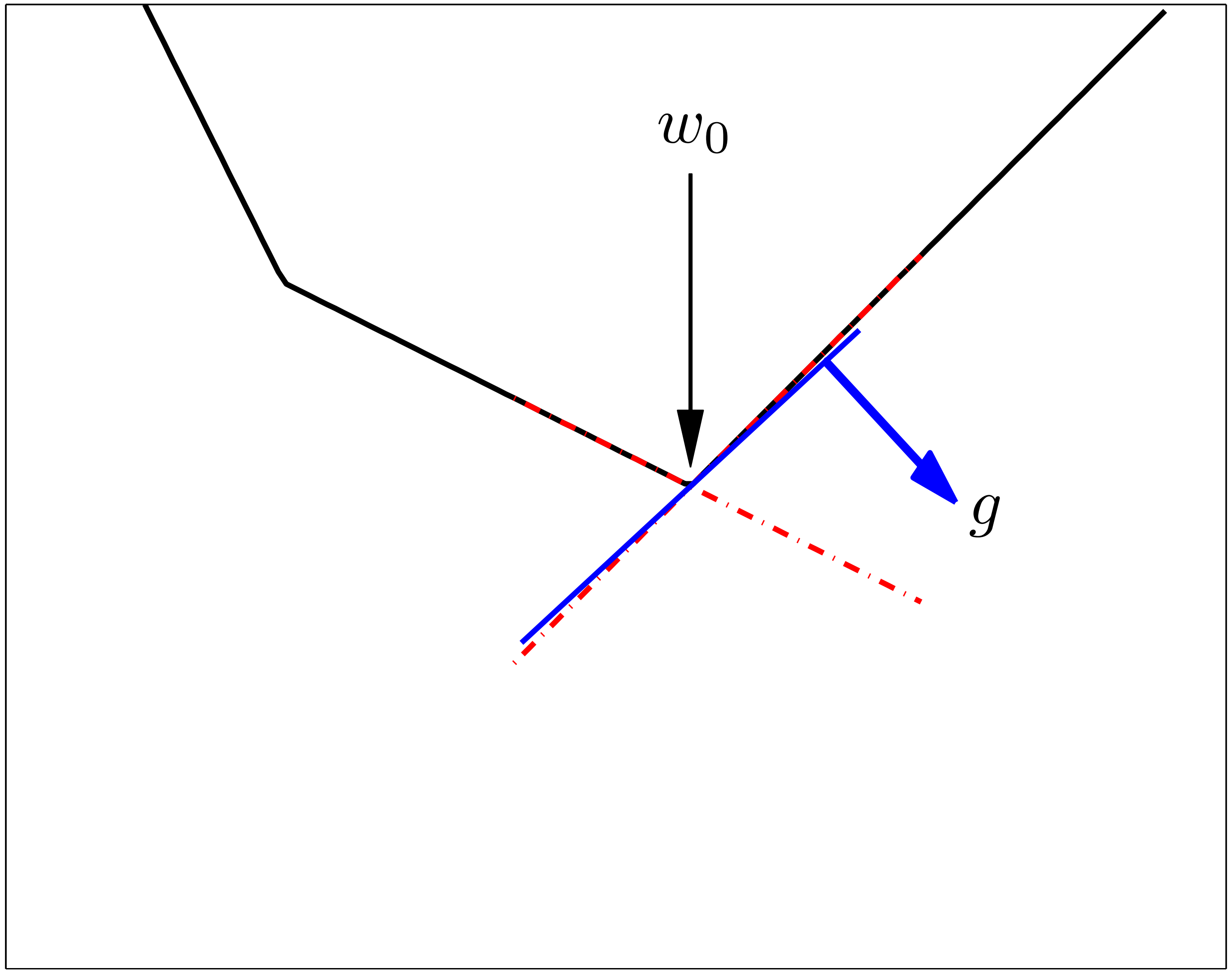
- Save Memory
- Easy to re-learn

How to update?

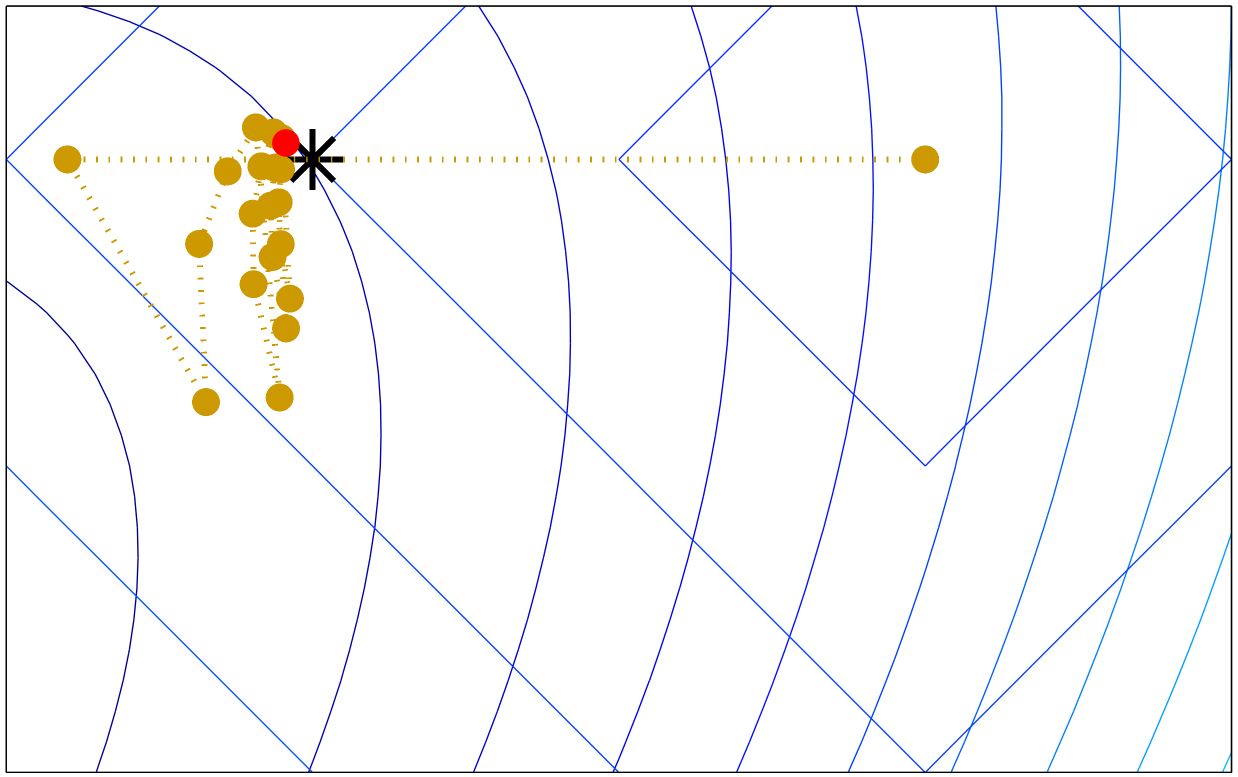
That is the question

The Fobos Algorithm

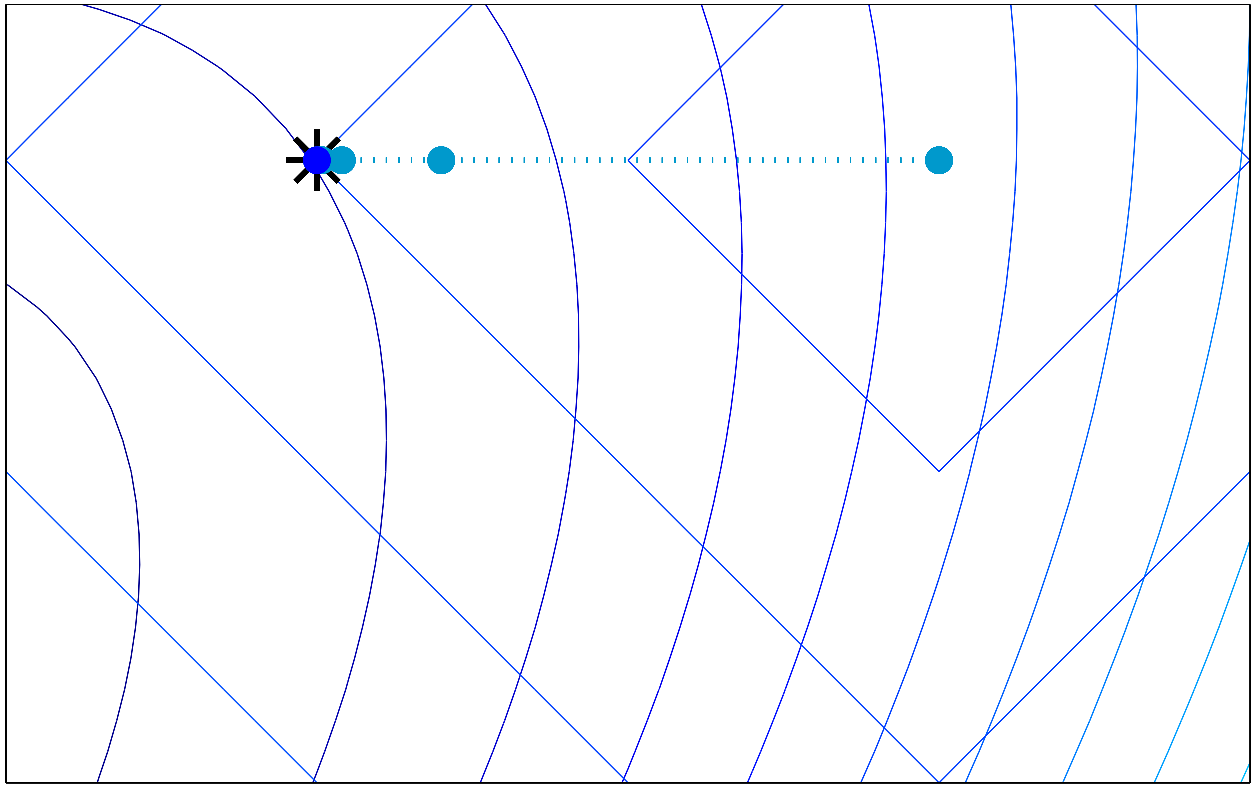
Classical method: Subgradient method



The Fobos Algorithm



Subgradient



Fobos

The Fobos Algorithm

Classical Methods: $w_t \rightarrow w_{t+1}$

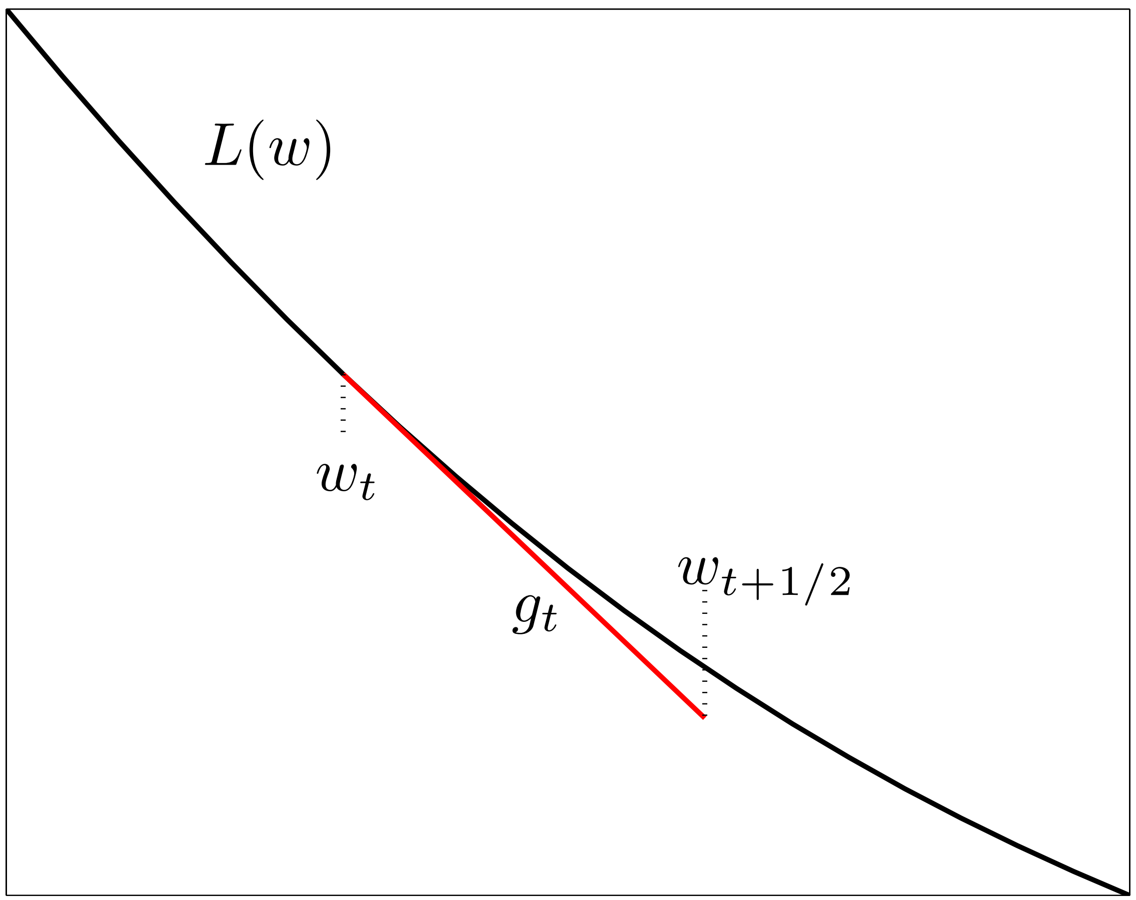
Fobos Algorithm: $w_t \underset{\text{step 1}}{\rightarrow} w_{t+\frac{1}{2}} \underset{\text{step 2}}{\rightarrow} w_{t+1}$

Fobos is an online learning algorithm specially for ℓ_1-regularized problems

The Fobos Algorithm

Step 1: minimize Loss

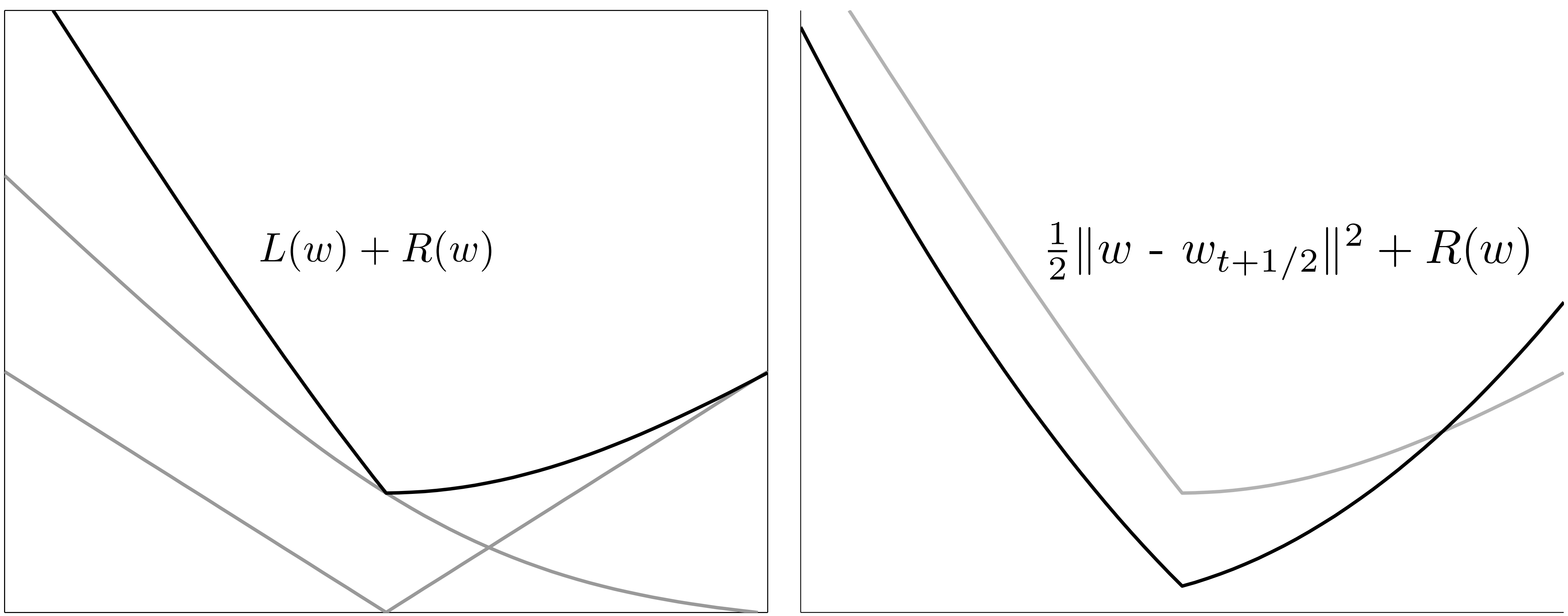
$$w_{t+\frac{1}{2}} = w_t - \eta_t g_t \text{ where } E[g_t] \in \partial L(w_t)$$



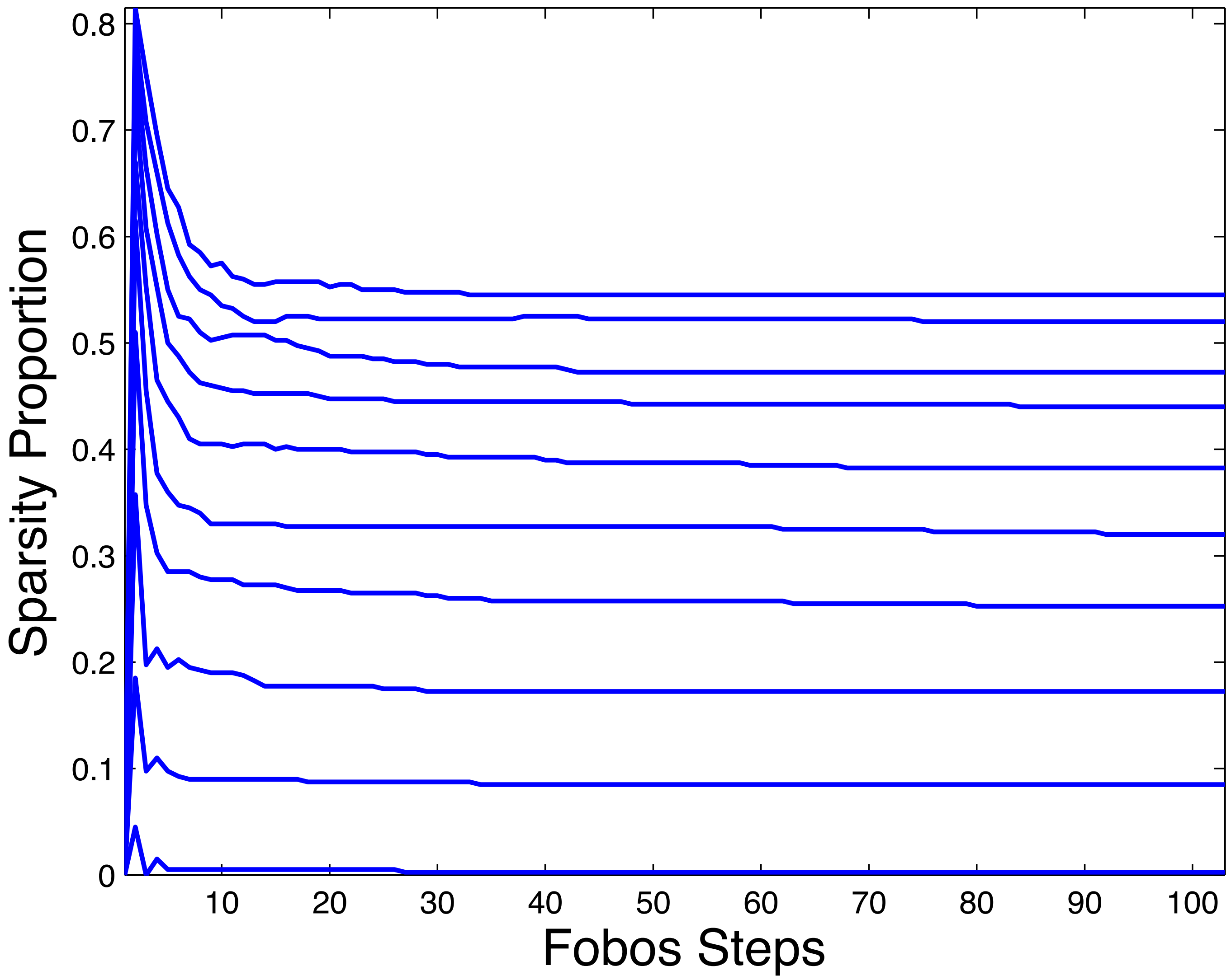
The Fobos Algorithm

Step 2: Regularization

$$w_{t+1} = \underset{w}{\operatorname{argmin}} \left\{ \frac{1}{2} \|w - w_{t+\frac{1}{2}}\|^2 + \eta_t R(w) \right\}$$

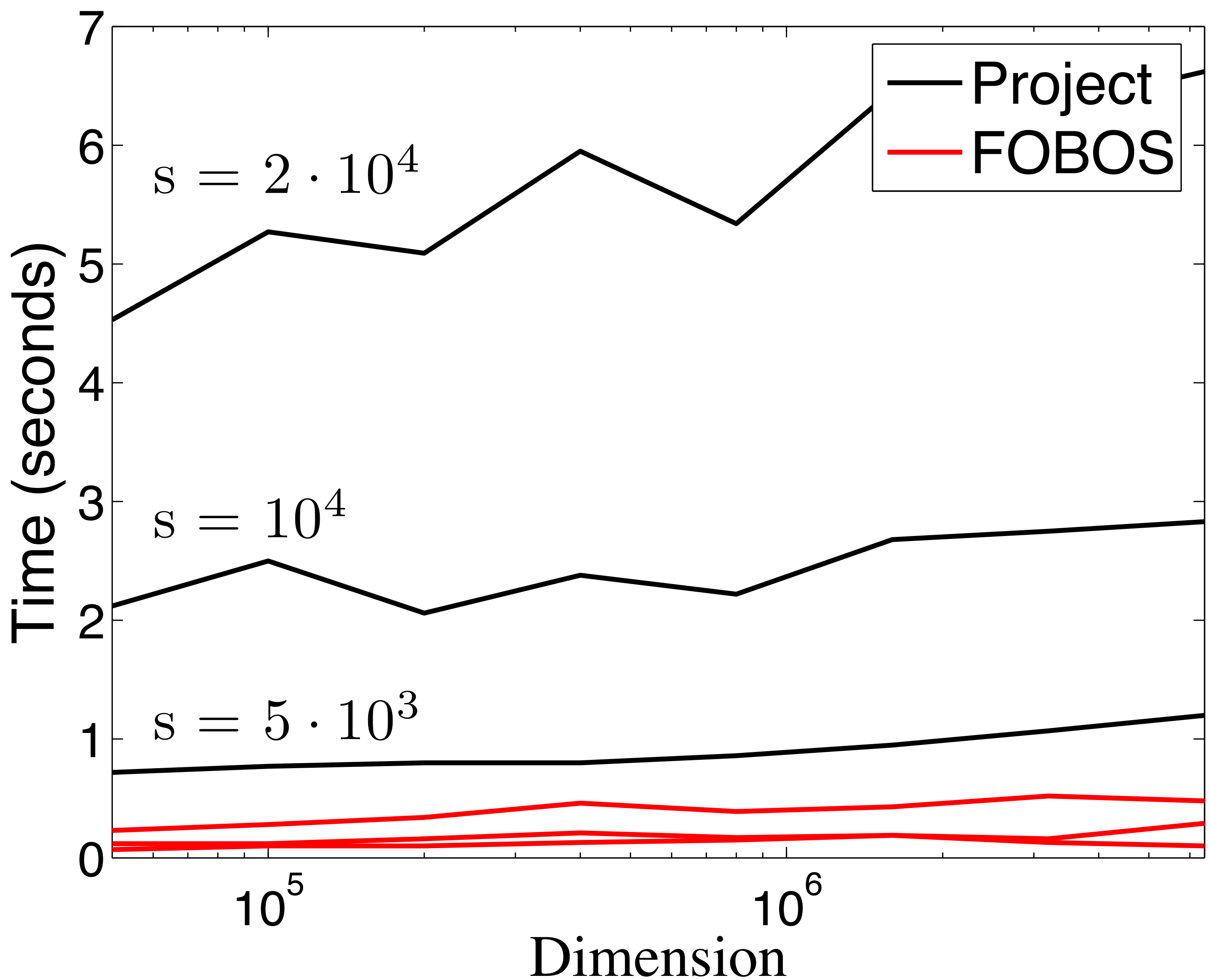


Sparsity



Sparsity as function of FOBOS steps on ℓ_1-regularized logistic regression

Sparse timing experiments



Comparison of ℓ_1-projection to FOBOS lazy update

Conclusions

- General framework for stochastic gradient with regularization.
- Lazy updates for efficency in high dimensions
- Fobos is efficient for online learning with sparse data

In my opinion

- The approach of Fobos to Forward-Backward Splitting is interesting.
- It should be faster if put structural assumptions of problem in it.